BLE

Maximizing BLE Throughput on iOS and Android

April 3, 2016 | By: Punch Through

Still confused about how fast you can really send data over Bluetooth Low Energy?

It's not just you. A google search will turn up many incomplete answers. Each OS and device has it's own restrictions, such as connection interval and number of packets per interval. For example, **did you know that with iOS you can send 4 packets per interval, and that HID devices can communicate 2.6x faster?**

If you want to finally clear up how fast you can communicate over BLE with an iPhone, Android device, or Mac, read on.

NOTE: This is Part 1 in Punch Through's Maximizing BLE Throughput blog series. Due to improvements to the Bluetooth Specification, hardware and mobile OSes, some aspects of this blog may be outdated. Here are links to our more recent posts in this series:

- **Part 2: Maximizing BLE Throughput: Use a Larger ATT MTU**
- **Part 3: Maximizing BLE Throughput: Data Length Extension (DLE)**

Let's start with the obvious wrong

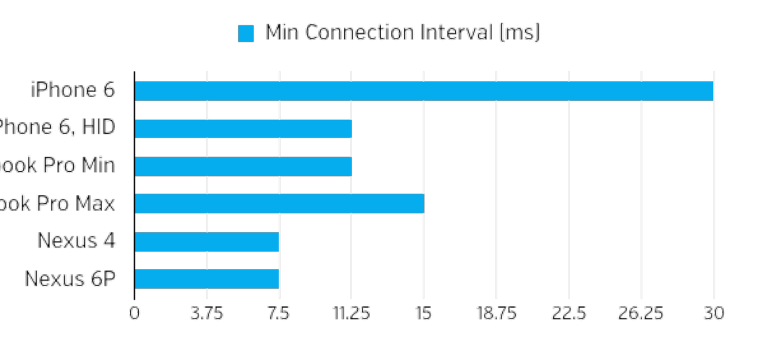
1 Mega bits per second is just wrong. That is the modulation frequency used to transfer a given BLE packet. Assuming this to be the maximum throughput is simply ignoring the communication protocol completely. There are many elements that affect/limit BLE's throughput such as:

– Maximum packet length
– Time delays in between packets
– Overhead bytes in a packet such as packet length, data integrity check, and general packet information.

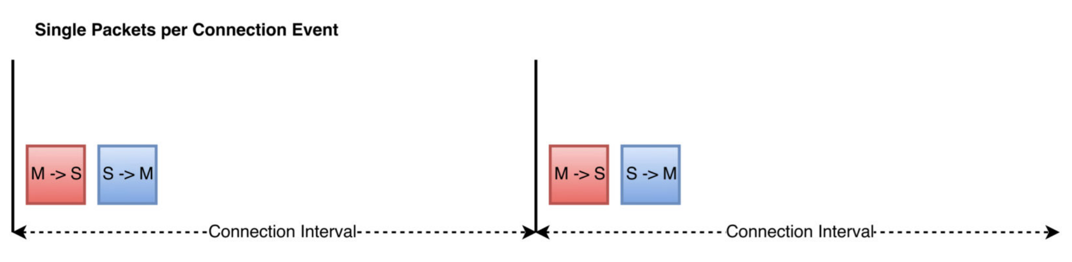
So, how about 50% to 75% of 1mbps? That is still 31.25 to 62.5 kB per second. That is still really good, so is it possible? Optimistically, maybe. Let's talk how that is a maybe.

The Science of BLE Throughput

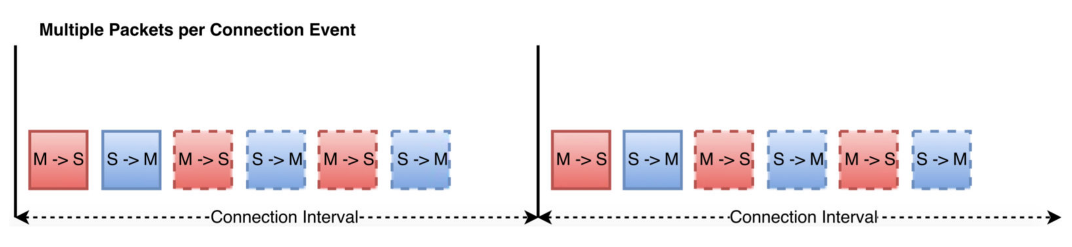
A BLE connection consists of only two devices, where one is called central (like master) and the other called peripheral (like slave). Each side communicates with the other on a given period called **Connection Interval** (minimum of 7.5 milliseconds and increases in steps of 1.25 milliseconds). Each instance of communication between two devices is called a **Communication Event**.



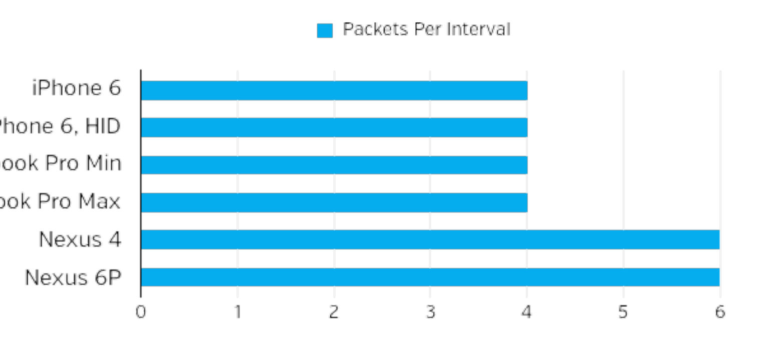
A communication event starts by the central (master) transmitting a packet while the peripheral device is in receive mode. If the peripheral received the packet from central successfully, the peripheral will subsequently transmit its packet while the central is in receive mode. By default, the central and peripheral each will transmit a packet even if they have nothing to send, this is often called **Empty Link Layer PD.**



For the cases when the central and peripheral have queued up messages, they can "ask" or "alert" the other side that they have more data to send before ending the connection event. This can result in multiple packets being exchanged during one interval. The connection event continues until a packet either fails to be received correctly, the sender is satisfied with ending the connection event, or, in a very unlikely scenario, the end of the interval has been reached.



The note to be learned is that in a connection event you can exchange more than two packets, which increases the throughput tremendously. It is important to know that maximum number of packets per connection event is dependent on the BLE stack/chipsets and is limited to 4 packets per connection event with iOS, and 6 packets per connection event in Android.



The equation for number of BLE packets in a given second boils down to the following:

$$\text{Packets Per Second} = \frac{1000 \text{ mSecs} * \text{Number of Packets in a Connection Interval}}{\text{Connection Interval mSecs}}$$

General BLE Data Throughput Formula

Remember, we stated that a given data throughput number in BLE is only valid when posted with context. Well, this context is the following three important elements:

- **Connection Interval** – How often the devices talk.
- **Number of Packets per Connection Interval** – How many packets exchanged when they do talk.
- **Packet Length** – Length of Application Data in a BLE packet.

The theoretical maximum application data throughput in BLE can be described as the following formula:

$$\text{Throughput} = \text{Packets Per Second} * \text{Data Per Packet}$$

Filling in the previous formula for Packets Per Second, we get the following:

$$\text{Throughput} = \frac{1000\text{mSecs} * \text{Number of Packets in a Connection Interval} * \text{Data Per Packet}}{\text{Connection Interval (mSecs)}}$$

Data Per Packet

In a forthcoming blog post, we will go in depth about the BLE packet structure and the possible sizes of application data in a packet, but for now it is safe to say that a BLE packet contains a **maximum of 20 bytes of application data** for BLE v4.0 and v4.1 (v4.2 includes a data packet length extension, but this is not used in iOS or Android yet). Remember the actual BLE packet is longer in size but naturally it includes segments for many important purposes such as CRC and headers for different layers.

Solving the Throughput Formula

If we remember anything from high school or middle school math, and that is dependent on which country you went to school, we learned that we cannot solve an equation with three unknowns, and that is the moral of the BLE throughput story. BLE throughput requires context for it to be meaningful and the context starts by determining the connection interval (the smaller generally gets you faster throughput, though there are some interesting exceptions) and the number of packets in a connection event. For simplicity, let's assume number of application data bytes per BLE packet is a constant 20 bytes.

For iOS 9.2 and iPhone 6, The minimum connection interval is 30 milliseconds unless you support HID over GATT Profile which allows for connections down to 11.25 milliseconds.

For Android devices, the minimum connection interval is 7.5 milliseconds

The "Bluetooth Accessory Design Guidelines for Apple Products" does mention a minimum interval of 20ms, but this is not seen in practice. In our experience, we have observed 8 packets (4 in one direction) per connection event, which gives us an ideal throughput of

*BLE Data Throughput for a Single Direction with iPhone 6 = (1000 mSecs) * 20 * (4) / (30 mSecs) = 2,667.66 Bytes per second.*

and if HID over GATT Profile is supported:

*Throughput is then increased to = (1000 mSecs) * 20 * 4 / (11.25 mSecs) = 7,111.11 Bytes per second*

Hint: You have the ability to increase your length of application data in a packet to iPhone 6 by three bytes which can increase your throughput by 15%. In future blogposts, we will discuss this.

Achieving Faster Throughput

Simple answer is sometimes you are just limited by the constraints on the three parameters of the other BLE device even if your BLE device can support lower connection intervals or send more than 4 packets per connection event.

For example, let's take the same peripheral device that we used to connect to iPhone 6 and connect to a different client (hardware) at a connection interval of 10mSecs and up to 12 packets (6 in one direction) per connection event. This will increase the ideal maximum as follows:

*Throughput for a Single Direction = (1000mSecs) * 20 * (6) / (10 mSecs) = 12,000 Bytes per second.*

Common Practices and Obstacles

In order to achieve the theoretical maximum throughput values, the system and the design of the devices must seek to use all the available time slots in a connection event to transfer a given data payload.

For example, if you perform a loopback for the data, you may find that the throughput measured on the application layer is well below the maximum value. This could be the result of many factors, such as:

Connection Interval

As a peripheral/slave, you can request for the connection interval to be lowered (if applicable) using a Connection Parameters Update Request message. Some Android devices may connect at slower connection intervals initially before lowering the connection interval when requested.

Interference

If interference exists and causes the first packet of the connection event to not be received correctly, the connection event is subsequently ended and the data will then have to be resent in the next connection event. Where as if the last packet of the connection even is interfered with, the repercussion is much less.

Importance of Number of Packets per Connection Event and Corner Cases

Let's assume the application's loopback payload to be 100 bytes long and the maximum number of packets per connection event is 4. This results in two connection events to transfer 100 bytes (ideally) in one direction. And it takes two connection events to transfer the data back.

Let's increase the application loopback payload to 140 bytes, which is segmented into 7 packets. That still results in two connection events to transfer the data.

Interestingly enough, the loopback application will yield similar throughput numbers for payload sizes of 100 and 140 bytes. It is import for the Bluetooth devices to use every time slot available in a connection interval to increase the data transfer.

Latency between the Application Layer and Physical layer

This is the time between when the application layer submits a packet to be transferred to the time that the packet is sent. In many platforms such as smart phones, this is difficult to optimize as the smart-phone may introduce a latency that cannot be removed; however, in most of the embedded systems, the developer should seek to find ways to remove this delay, whether it is by processing the packets faster or methods suggested by the BLE Stack/Chipset provider.

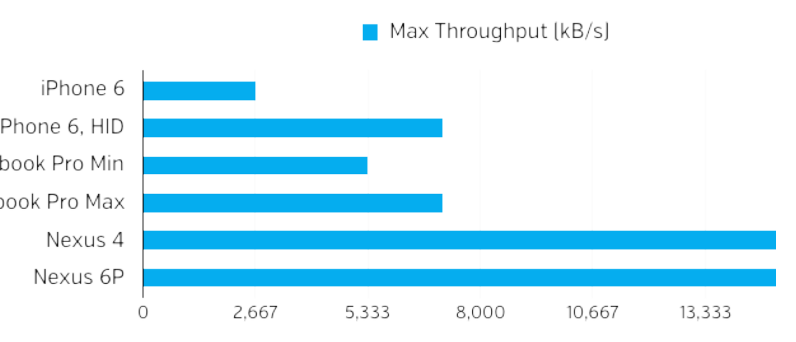
For example, in the loopback throughput measurement if a device receives 160 bytes of data in two connection events but it fails to process the payload quickly for the next connection event, then you will be penalized with a connection event that has gone untapped.

Proper Methods to Use When Transferring Application Data

Transferring data to and from the GATT can be achieved in multiple ways. In order to achieve higher throughput, the method that allows for more than one packet in a connection event needs to be selected. Use **Write Command** instead of **Write Request** or **Blob Write** when sending data from GATT Client to GATT Server, Use **Notification** instead of **Indication** when sending data from GATT Server to GATT Client. We will discuss the pros and cons of the different GATT data transfer methods in a separate blog post.

Maximum Throughput with Smartphones, Laptops

Here are maximum throughput numbers for a given set of popular smart phones and laptops:



iPhone 6,6+,6S,6S+:
Normal Connection Interval of 30mSecs: 2,667 bytes/sec
Connection Interval for HID Over GATT is Present 11.25mSecs: 7,111 bytes/sec

MacBook Pro – OS X (Varies on models):
Maximum Connection Interval range of (11.25 – 15mSecs): 7,111 bytes/sec – 5334 bytes/sec

Android
Nexus 4: Minimum Connection Interval of 7.5 mSecs – Packets per connection Interval: 6 – Throughput: 16,000 Bytes/Sec
Nexus 6P: Minimum Connection Interval of 7.5 mSecs – Packets per connection Interval: 6 – Throughput: 16,000 Bytes/Sec

This is part 1 of a 2 part series. You can find part 2 here: Maximizing BLE Throughput Part 2 – Choosing a larger MTU.

Punch Through

Learning and collaboration efforts from current and past employees and guest authors of Punch Through.

RECENT POSTS

Meet the Team – Tyler Grunenwald
Erin Moore

Mastering all 5 Kotlin Scope Functions
Chee Yi Ong

How to Delete Bonds with the Nordic nRF5 SDK
Amanda Christiansen

Meet the Team – Matt Kruger
Emily Hinderaker

CONTACT
MINNEAPOLIS
123 N. 3rd St,
Suite 500
Minneapolis, MN
55401
info@punchthrough.com